AMENDMENTS TO THE CLAIMS

Please amend the claims as follows.

1. (Currently Amended) A method for obtaining data from a kernel, comprising:

    obtaining, by a first processor, first data from the kernel using one of a plurality of probes in the kernel; [[and]]

    applying an aggregation function to the first data to obtain a first intermediate result;

    storing the [[data]] first intermediate result in a first data set in a[[n]] first aggregation buffer using an aggregation function, wherein the first data set comprises an aggregation identifier that identifies the aggregation function;

    obtaining, by the first processor, second data from the kernel using one of the plurality of probes;

    applying the aggregation function to the second data and the first intermediate result to obtain a second intermediate result;

    replacing the first intermediate result with the second intermediate result in the first data set in the first aggregation buffer;

    obtaining, by a second processor, third data from the kernel using one of the plurality of probes in the kernel;

    applying the aggregation function to the third data to obtain a third intermediate result;

    storing the third intermediate result in a second data set in a second aggregation buffer, wherein the second data set comprises the aggregation identifier;

    obtaining, by the second processor, fourth data from the kernel using one of the plurality of probes;

    applying the aggregation function to the fourth data and the third intermediate result to obtain a fourth intermediate result;

    replacing the third intermediate result with the fourth intermediate result in the second data set in the second aggregation buffer;

    generating an aggregation result for the kernel by applying the aggregation function to the third intermediate result and the fourth intermediate result; and

storing the aggregation result in a user-level buffer,

wherein the first aggregation buffer and the second aggregation buffer are kernel-level buffers.

2. (Cancelled)

3. (Currently Amended) The method of claim 1, wherein ~~obtaining~~ replacing the first intermediate result [[data]] comprises:

obtaining an expression, ~~a new value~~ the first intermediate result, and [[an]] the aggregation identifier, and

generating a key using the expression and the aggregation identifier,

wherein the key is used to locate the first data set and wherein the first data set comprises the key.

4. -11. (Cancelled)

12. (Currently Amended) A method for integrating data into a user-level table, comprising:

> obtaining a first data set from a[[n]] first aggregation buffer, wherein the first data set comprises a key component, an aggregation identifier component, and a value component, wherein the value component comprises a first intermediate aggregation result obtained by applying an aggregation function to data obtained during execution of a first processor, wherein the aggregation identifier component comprises an aggregation identifier that identifies the aggregation function, and wherein the first aggregation buffer is a kernel-level buffer associated with the first processor;

> obtaining [[an]] the aggregation identifier and using the aggregation identifier ~~matching a value of the aggregation identifier in the aggregation identifier component of the data set~~ to obtain a user-level table key;

> hashing the user-level table key to obtain a generated hash key;

> locating a user-level table entry, in the user-level table comprising a hash key matching the generated hash key; and

> updating a current value of a value component of [[a]] the user-level table entry to obtain a new value ~~if a user-level table entry having a hash key matching the generated hash key is found~~, wherein updating the current value component comprises applying [[an]] the aggregation function ~~corresponding to the aggregation identifier~~ to the current value and the first intermediate aggregation result to obtain the new value ~~component using the new value as input~~;

> ~~creating a new user-level table entry if a user-level table entry having a hash key matching the generated hash key is not found, wherein creating the new user-level table entry comprises storing the generated hash value, and an initial value in a value component of the new user-level table entry in the new user-level table entry; and~~

> ~~updating the value component in the new user-level table entry, wherein updating the value component in the new user-level table entry comprises applying the aggregation function corresponding to the aggregation identifier to the value component in the new user-level table entry using the value component of the data set as input.~~

wherein the current value is a second intermediate result obtained by applying the
aggregation function to data obtained during execution of a second processor,

wherein the second intermediate result is stored in a second data set comprising the
aggregation identifier,

wherein the second intermediate result is stored in a second aggregation buffer prior to being
stored in the value component, and

wherein the second aggregation buffer is a kernel-level buffer associated with the second
processor.

13. (Cancelled)

14. (Currently Amended) The method of claim 12, wherein obtaining the aggregation identifier
comprises ~~matching the value of the aggregation identifier comprises~~ searching at least one
selected from the group consisting of a user-level dictionary and a kernel level dictionary.

15. – 23. (Cancelled)

24. (Currently Amended) A computer system on a network obtaining data from a kernel comprising:

a processor;

a memory;

a storage device; and

software instructions stored in the memory for enabling the computer system to:

obtain, by a first processor, first data from the kernel using one of a plurality of probes in the kernel; [[and]]

apply an aggregation function to the first data to obtain a first intermediate result;

store the [[data]] first intermediate result in a first data set in a[[n]] first aggregation buffer using an aggregation function, wherein the first data set comprises an aggregation identifier that identifies the aggregation function;

obtain, by the first processor, second data from the kernel using one of the plurality of probes;

apply the aggregation function to the second data and the first intermediate result to obtain a second intermediate result;

replace the first intermediate result with the second intermediate result in the first data set in the first aggregation buffer;

obtain, by a second processor, third data from the kernel using one of the plurality of probes in the kernel;

apply the aggregation function to the third data to obtain a third intermediate result;

store the third intermediate result in a second data set in a second aggregation buffer, wherein the second data set comprises the aggregation identifier;

obtain, by the second processor, fourth data from the kernel using one of the plurality of probes;

apply the aggregation function to the fourth data and the third intermediate result to obtain a fourth intermediate result;

replace the third intermediate result with the fourth intermediate result in the second data set in the second aggregation buffer;

generate an aggregation result for the kernel by applying the aggregation function to the third intermediate result and the fourth intermediate result; and

store the aggregation result in a user-level buffer,

wherein the first aggregation buffer and the second aggregation buffer are kernel-level buffers.